Opened: Friday, 15 March 2024, 6:00 AM
Due: Friday, 29 March 2024, 11:59 PM

This assignment is worth 9% of the overall weight.

In this assignment, you will implement a simplified version of the JPEG encoder and decoder. You can read about JPEG compression here and here.

Introduction:

The JPEG compression pipeline reduces the amount of data required to represent an image while minimizing the loss of perceptual quality. The major steps in the JPEG compression pipeline are as follows:

1. **Image transformation:** The input RGB image is is transformed to the YCbCr space.
2. **Image segmentation:** The YCbCr image is divided into 8x8 blocks of pixels (in each channel)
3. **Discrete Cosine Transform (DCT):** For each block, a 2D Discrete Cosine Transform (DCT) is applied, which converts the pixel values in the block into a set of frequency coefficients that represent the spatial frequency components of the image within that block.
4. **Quantization:** The frequency coefficients are then quantized, which means they are divided by a quantization matrix. This process reduces the coefficients' precision and introduces some information loss. The quantization step is where most of the compression occurs.
5. **Encoding:** The quantized coefficients are then encoded using a lossless compression algorithm, such as Huffman encoding (beyond the scope of the syllabus). This step further reduces the amount of data required to represent the image.
6. **Decoding:** To reconstruct the image, the encoded data is decoded, and the quantized coefficients are multiplied by the original quantization matrix to obtain the approximate original frequency coefficients.
7. **Inverse DCT:** An inverse 2D DCT is applied to each block for each channel to obtain the reconstructed image in the YCbCr space.
8. **Image Transformation:** The YCbCr image is transformed back to the RGB space.

You are provided with one source file called **A5_submission.py**. You need to complete the functions and parts indicated in the source file. These have been marked with "TODO". You can add any other functions or other code you want to use but they must all be in the same file. You need to submit **only** the completed **A5_submission.py**.

NOTE: The source file includes a Zig-zag section at the very beginning (which should NOT be altered) that contains two functions for zig-zag and inverse zig-zag scanning of an image.

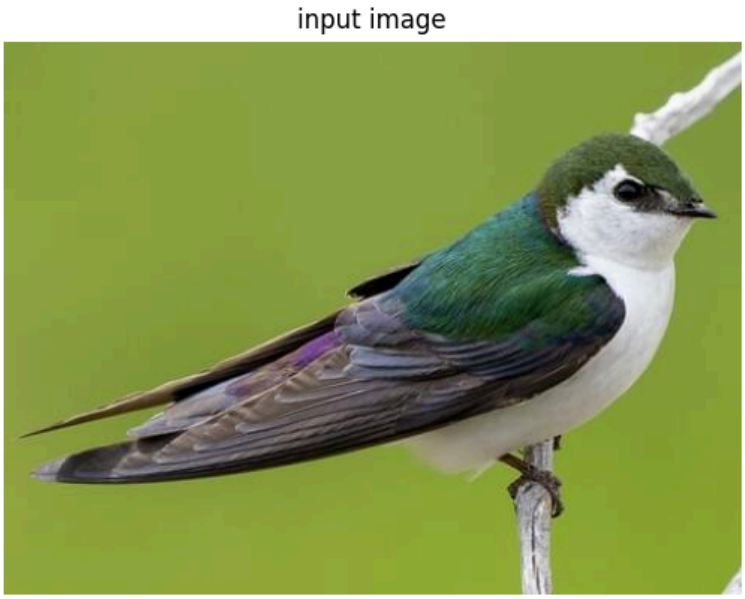
Part 1 (60%): JPEG encoding

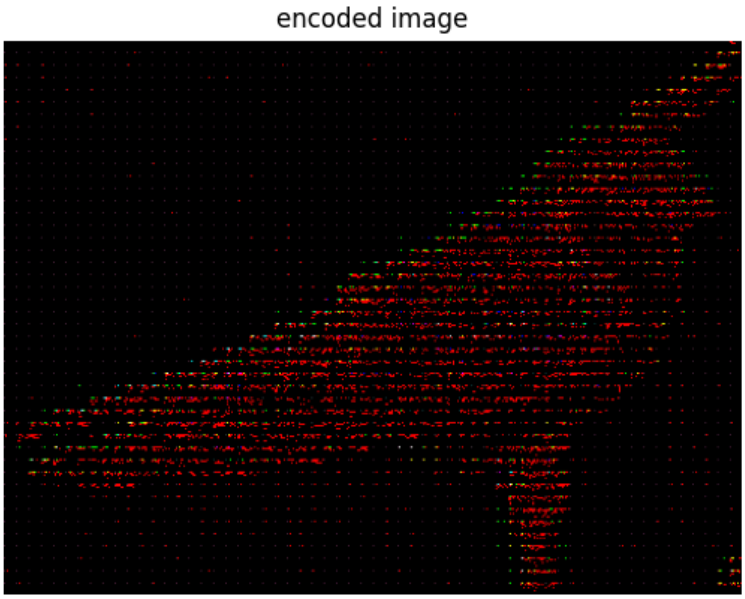
In this part, you will implement a JPEG encoder. Download **this image** as your input. You need to complete all the TODOs in this section. **A5_submission.py** has explanations of what you need to write.

The JPEG encoder section has four major steps:

1. Transforming the image from the RGB space to the YCbCr space (see **here**).
2. Creating a 2D discrete cosine transform (DCT) function called dct2d (you can use this **scipy function** and also **this one**)
3. Quantization: Process of removing the high-frequency data. You can read more about quantization **here**.
   NOTE: Refer to this **PDF** and use Fig. 1 as your quantization matrices (the upper table is used for Y channel and the lower table is used for Cb and Cr channels)
4. Degree of compression: The calculation for the degree of compression requires computing the number of pixel locations with zero values before and after quantization over the multiple blocks. You need to calculate the percentage of zero-pixel locatio only for the Y channel.

Expected outputs:






```
Percentage of non-zero elements in Luma channel:
Before compression:  49.536458333333336 %
After compression:  8.370949074074074 %
```

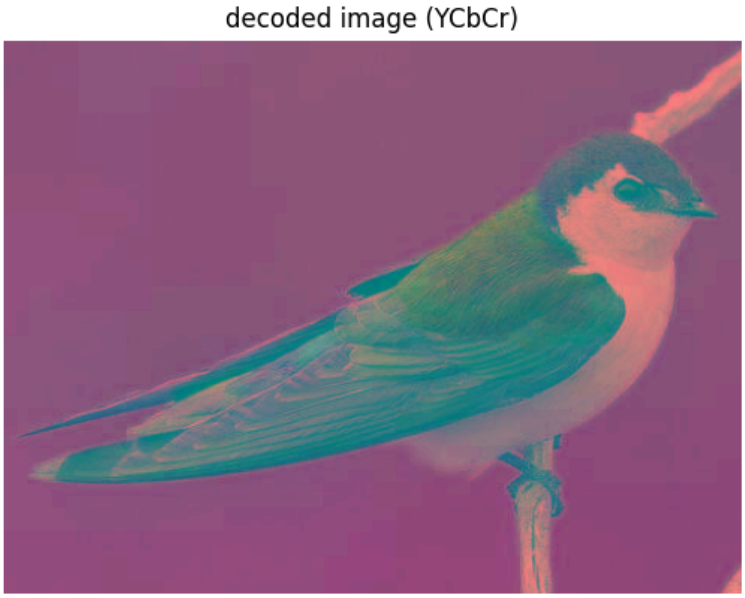
Part 2 (40%): JPEG decoding

In this part, you will implement a JPEG decoder. Download **this image** as your input. You need to complete all the TODOs in this section. **A5_submission.py** has explanations of what you need to write.
The JPEG decoder section has two major steps:

1. Creating a 2D discrete cosine transform (DCT) function called *idct2D* (you can use this **scipy function**)
2. De-quantization

3. Transform the image from YCbCr space back to the RGB space (see **here**)

Expected outputs:



Add submission

Submission status

Attempt number	This is attempt 1 (1 attempts allowed).
Submission status	No submissions have been made yet
Grading status	Not graded
Time remaining	9 days 9 hours remaining
Last modified	-
Submission comments	▸ Comments (0)